A Data and Model Visualization System for Android Malware Detection

CPSC 547

Michael Cao

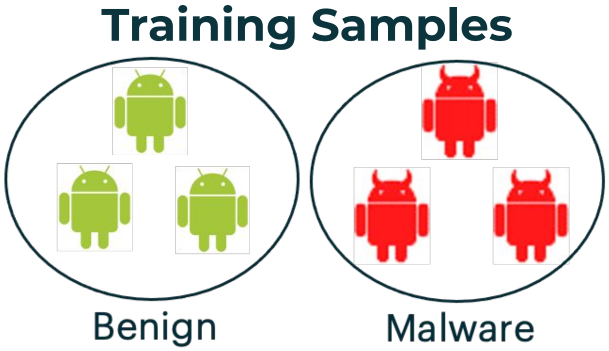
Gabby Xiong

Android Malware Detection

- Cybercriminals target mobile due to large user base
- Rely on machine learning

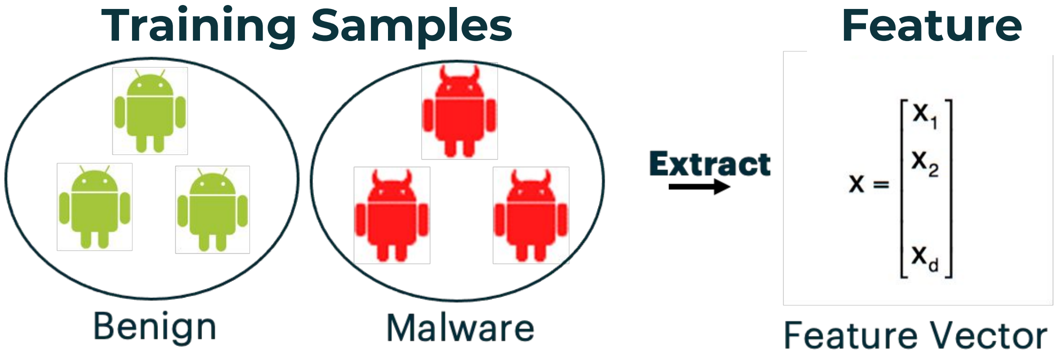
Android Malware Detection

- Cybercriminals target mobile due to large user base
- Rely on machine learning



Android Malware Detection

- Cybercriminals target mobile due to large user base
- Rely on machine learning



Training Samples

Benign

Malware

Extract

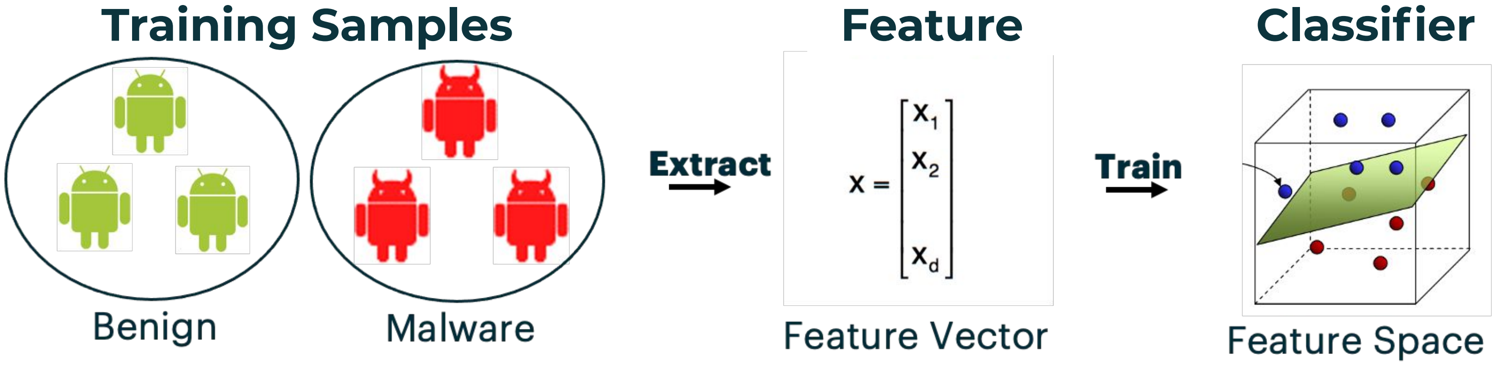
Feature

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \\ x_d \end{bmatrix}$$

Feature Vector

Android Malware Detection

- Cybercriminals target mobile due to large user base
- Rely on machine learning



Training Samples

Benign

Malware

Extract

Feature

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \\ x_d \end{bmatrix}$$

Feature Vector

Train

Classifier

Feature Space

Issue with Android Malware Detection

- Focus on producing models with high accuracy
    - What about model attackability?

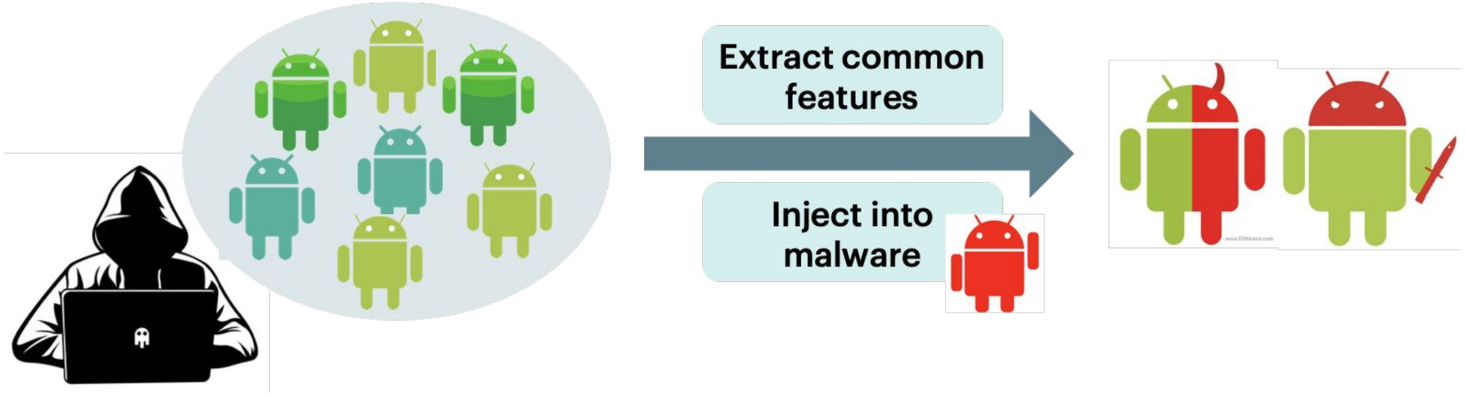
Issue with Android Malware Detection

- Focus on producing models with high accuracy
    - What about model attackability?

Attackability of the Model is also Important!

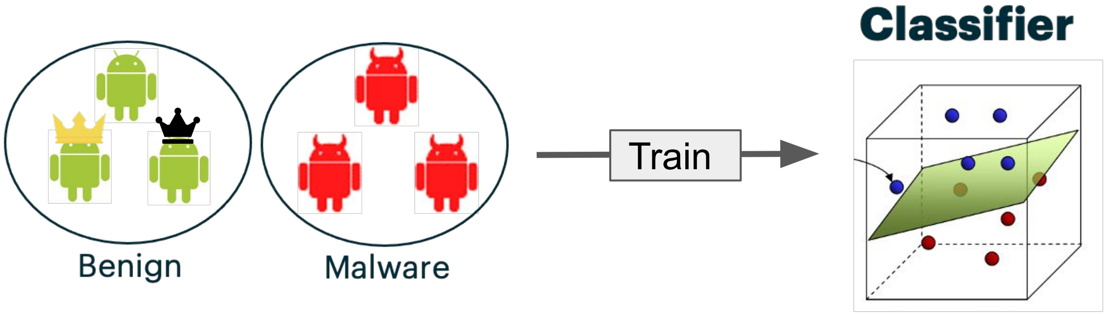
Issue with Android Malware Detection

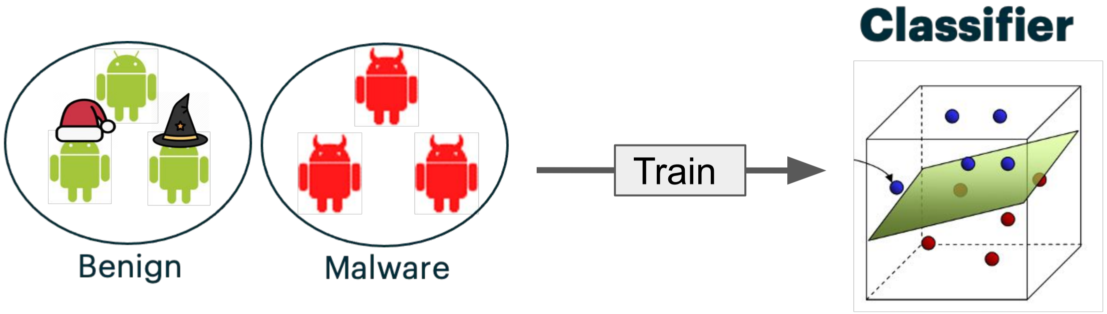
- Focus on producing models with high accuracy
    - What about model attackability?
- Mimicry Attack:
    - Inject features they think represent benign to mislead detection



Issue with Android Malware Detection

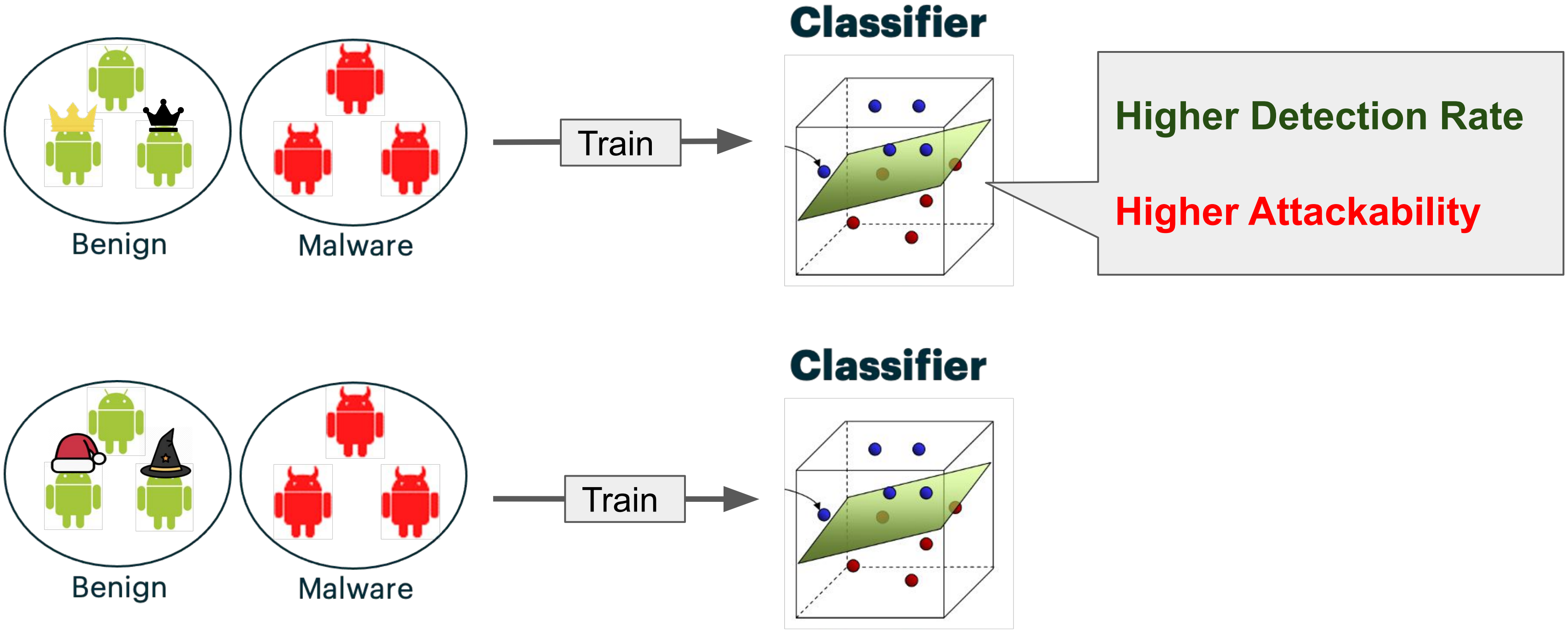
- Different sets of benign samples, different impacts on results





Issue with Android Malware Detection

- Different sets of benign samples, different impacts on results

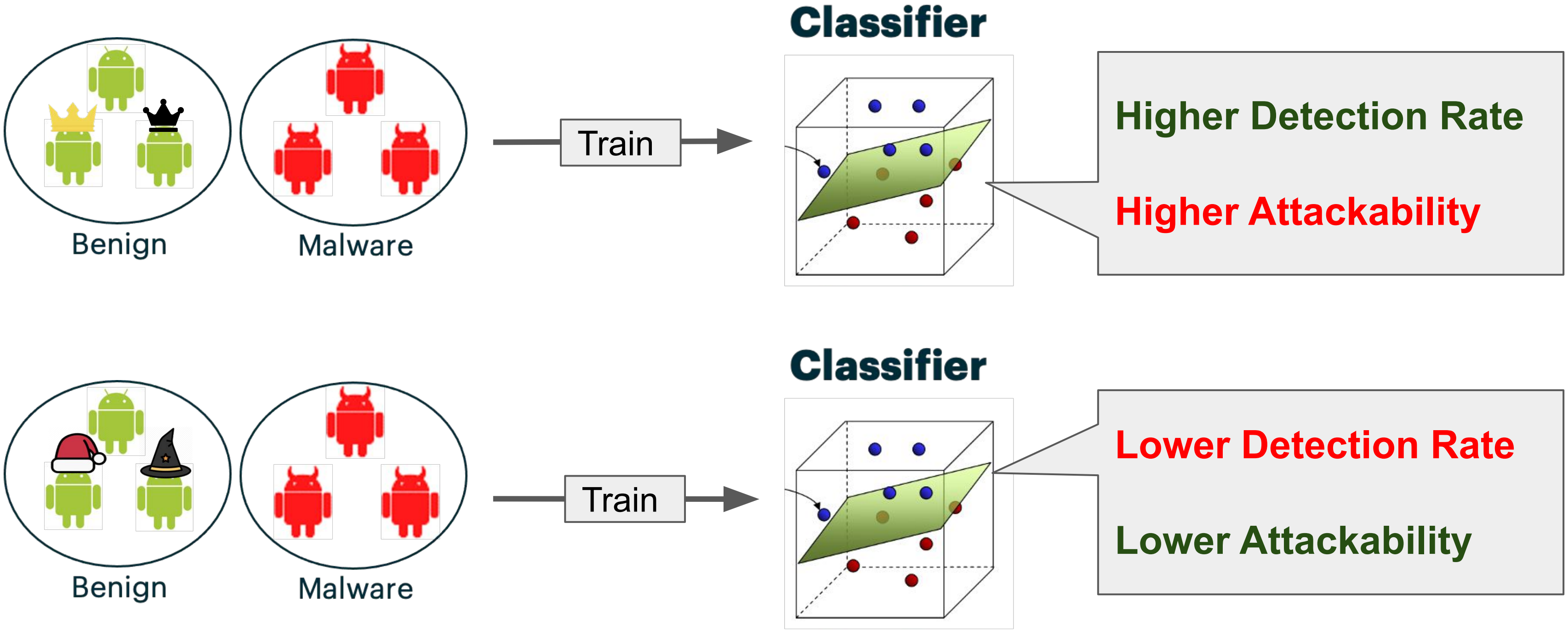


Higher Detection Rate

Higher Attackability

Issue with Android Malware Detection

- Different sets of benign samples, different impacts on results



Higher Detection Rate

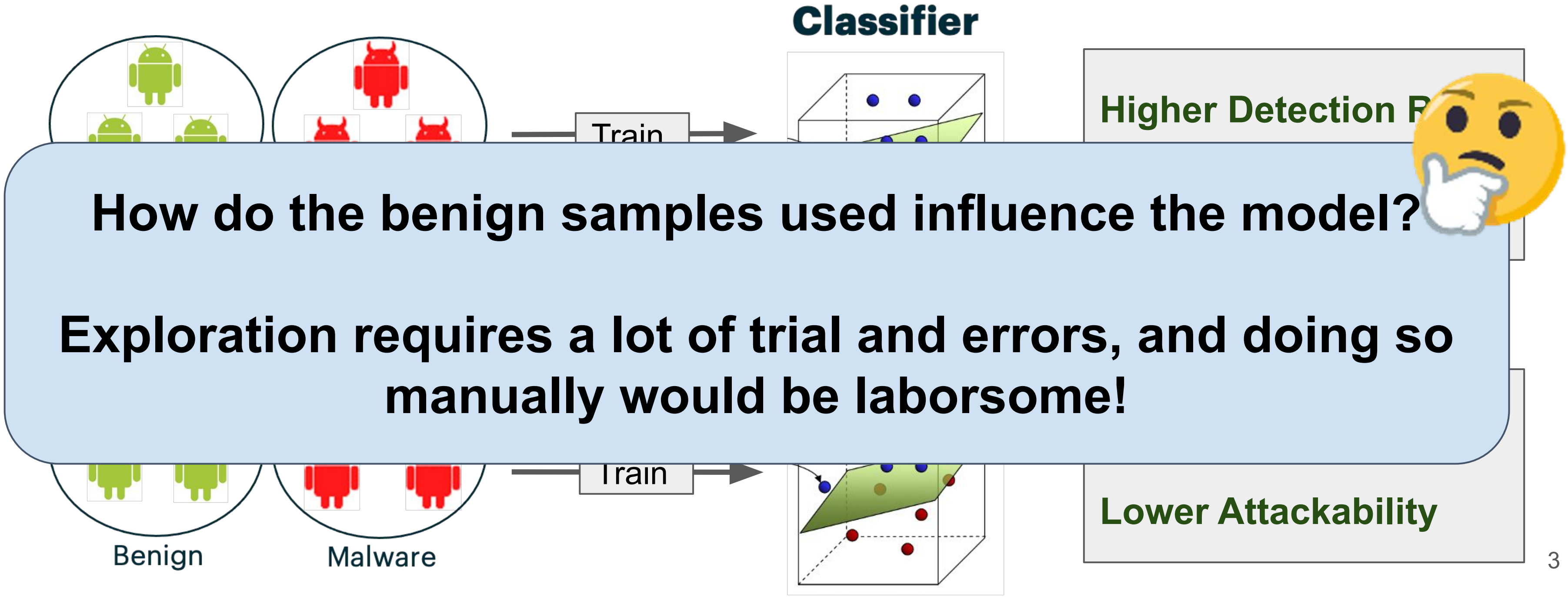
Higher Attackability

Lower Detection Rate

Lower Attackability

Issue with Android Malware Detection

- Different sets of benign samples, different impacts on results

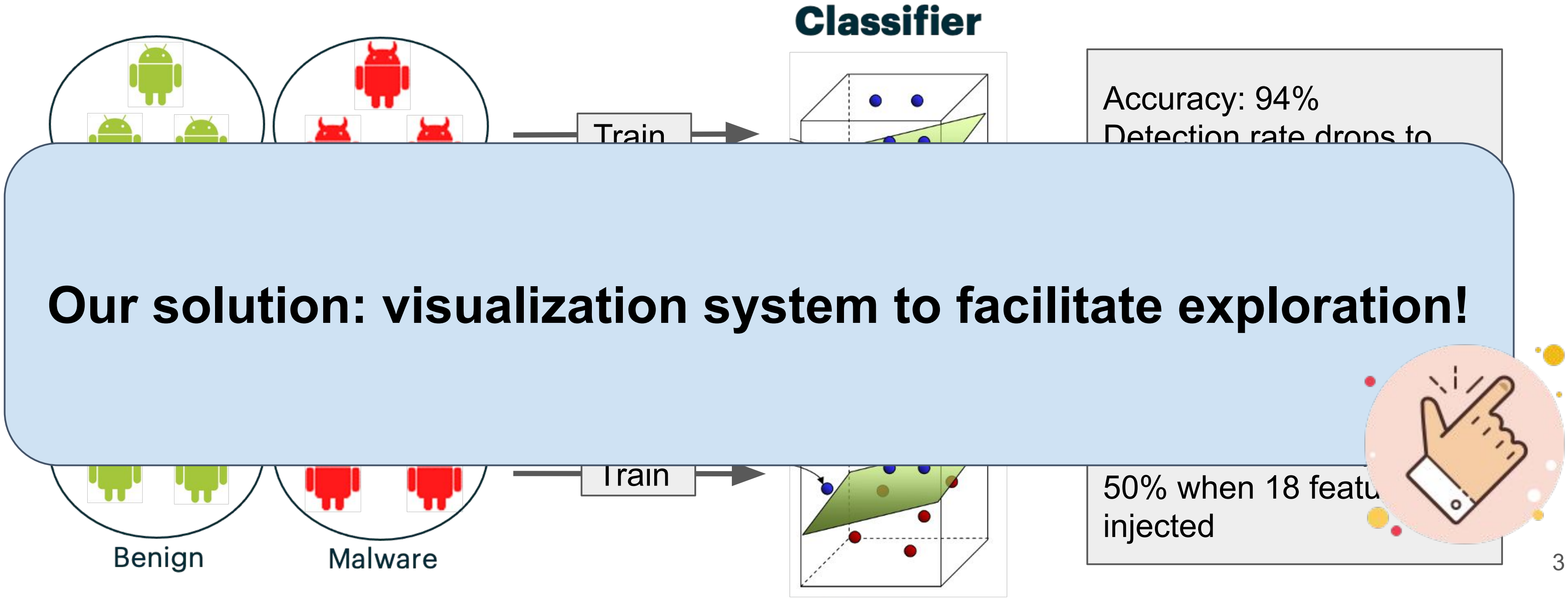


How do the benign samples used influence the model?

Exploration requires a lot of trial and errors, and doing so manually would be laborsome!

Issue with Android Malware Detection

- Different sets of benign samples, different impacts on results

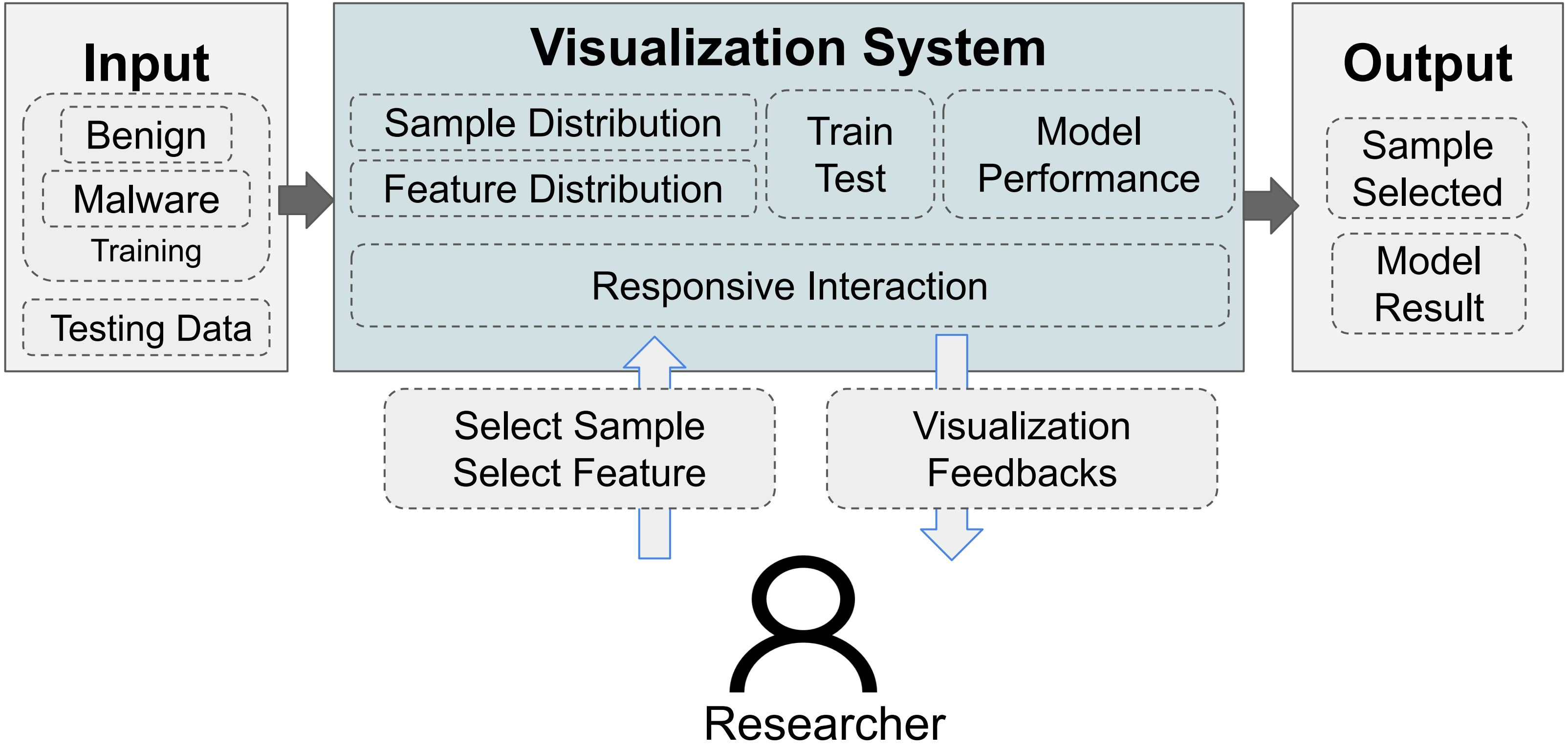


Accuracy: 94%
Detection rate drops to

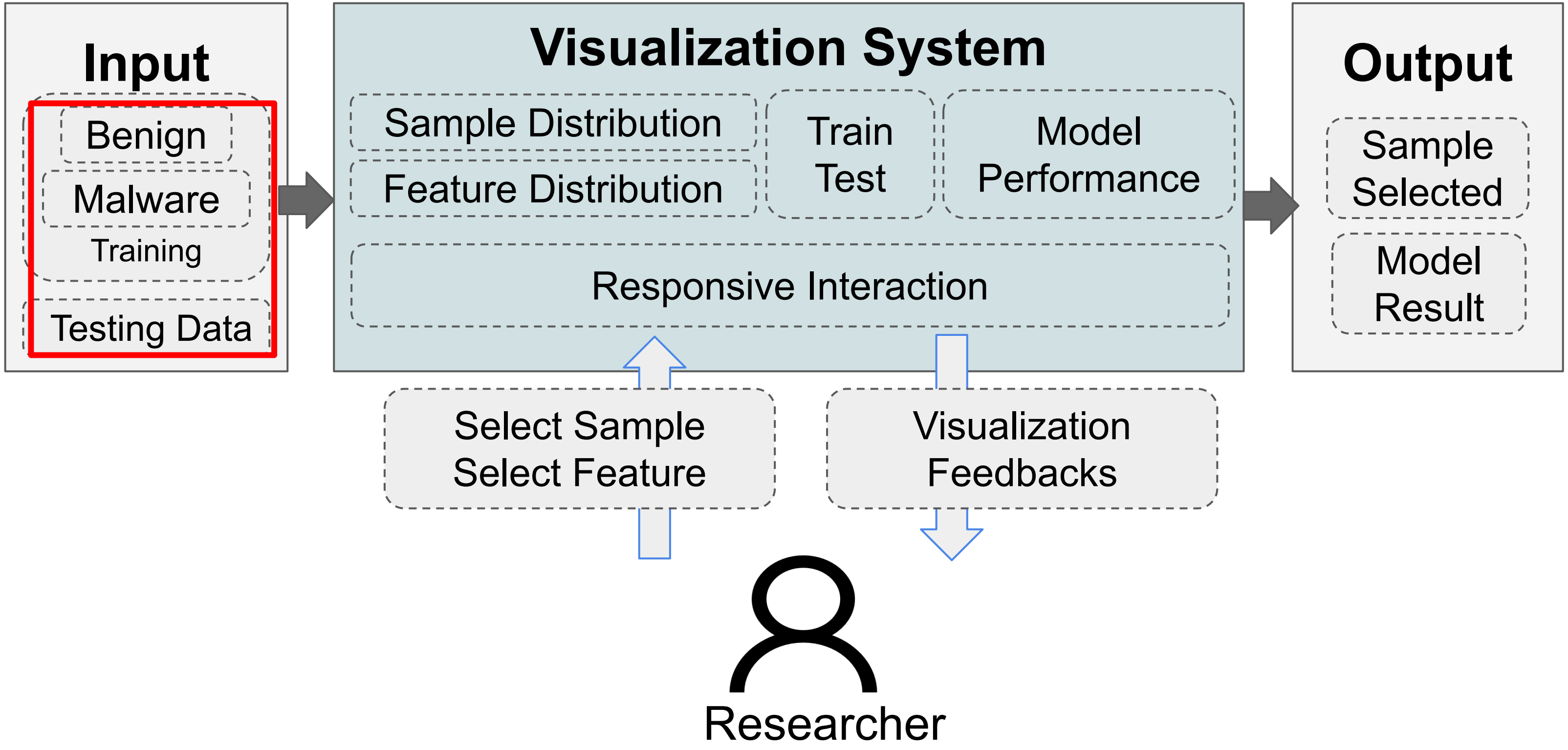
50% when 18 featu
injected

Our solution: visualization system to facilitate exploration!

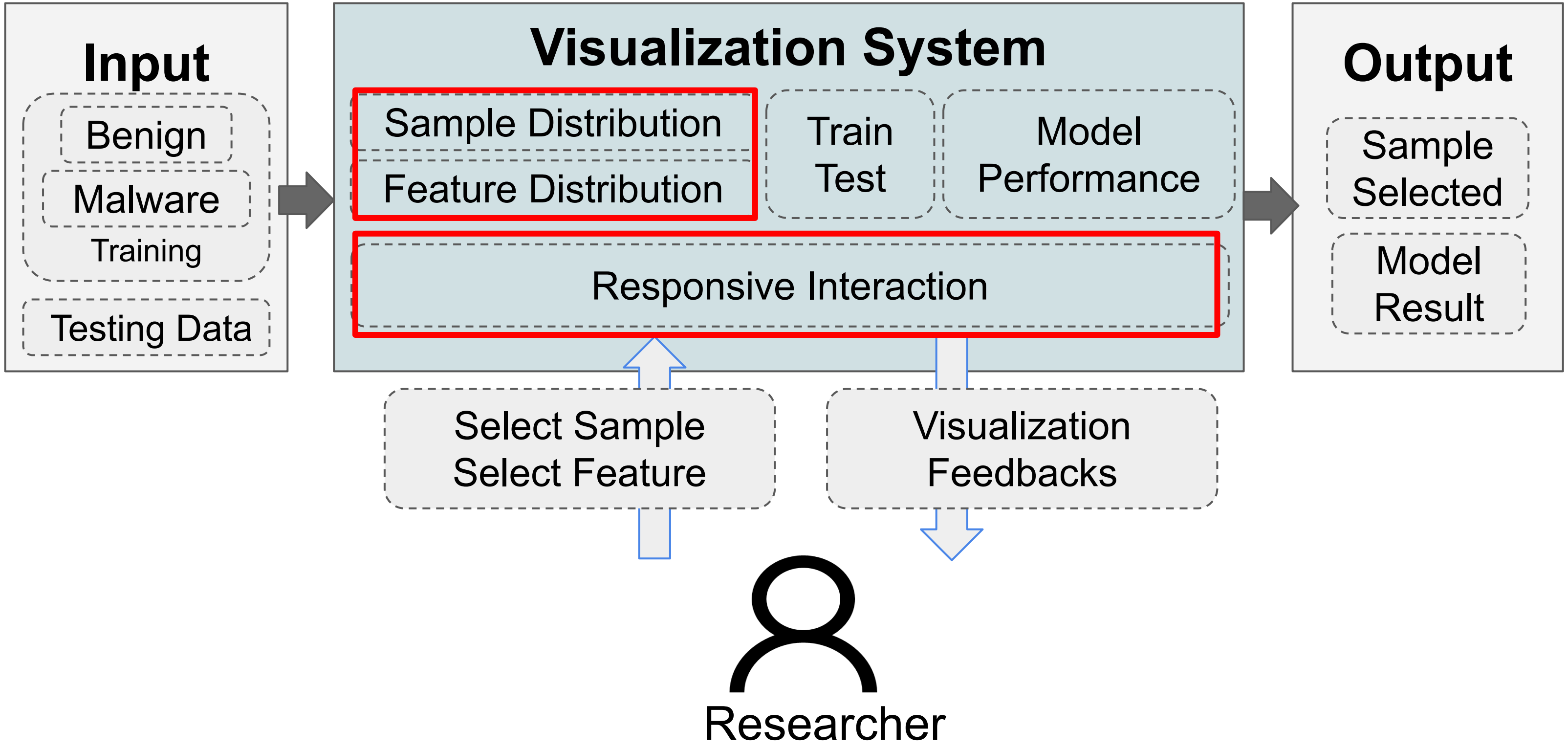
High Level Overview



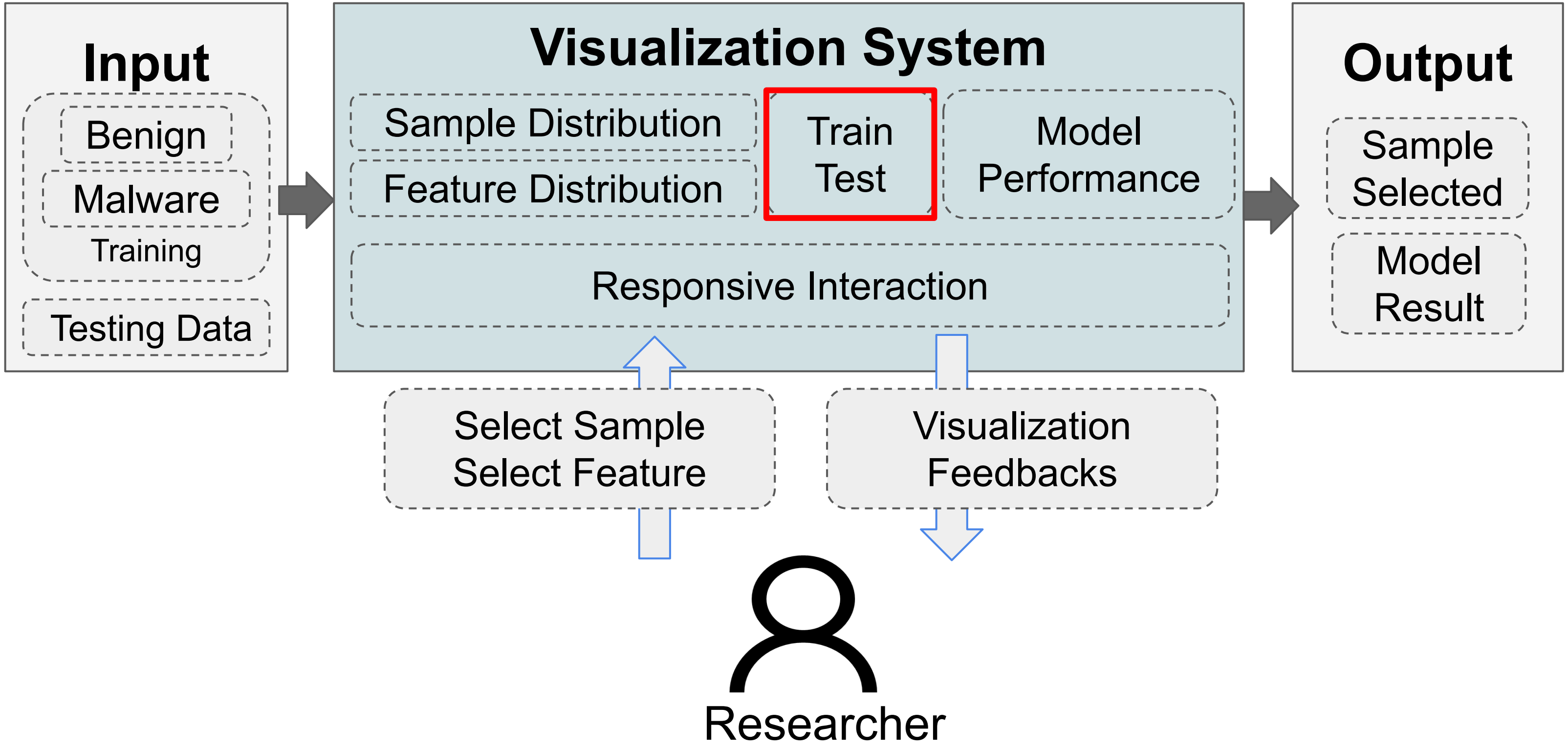
High Level Overview



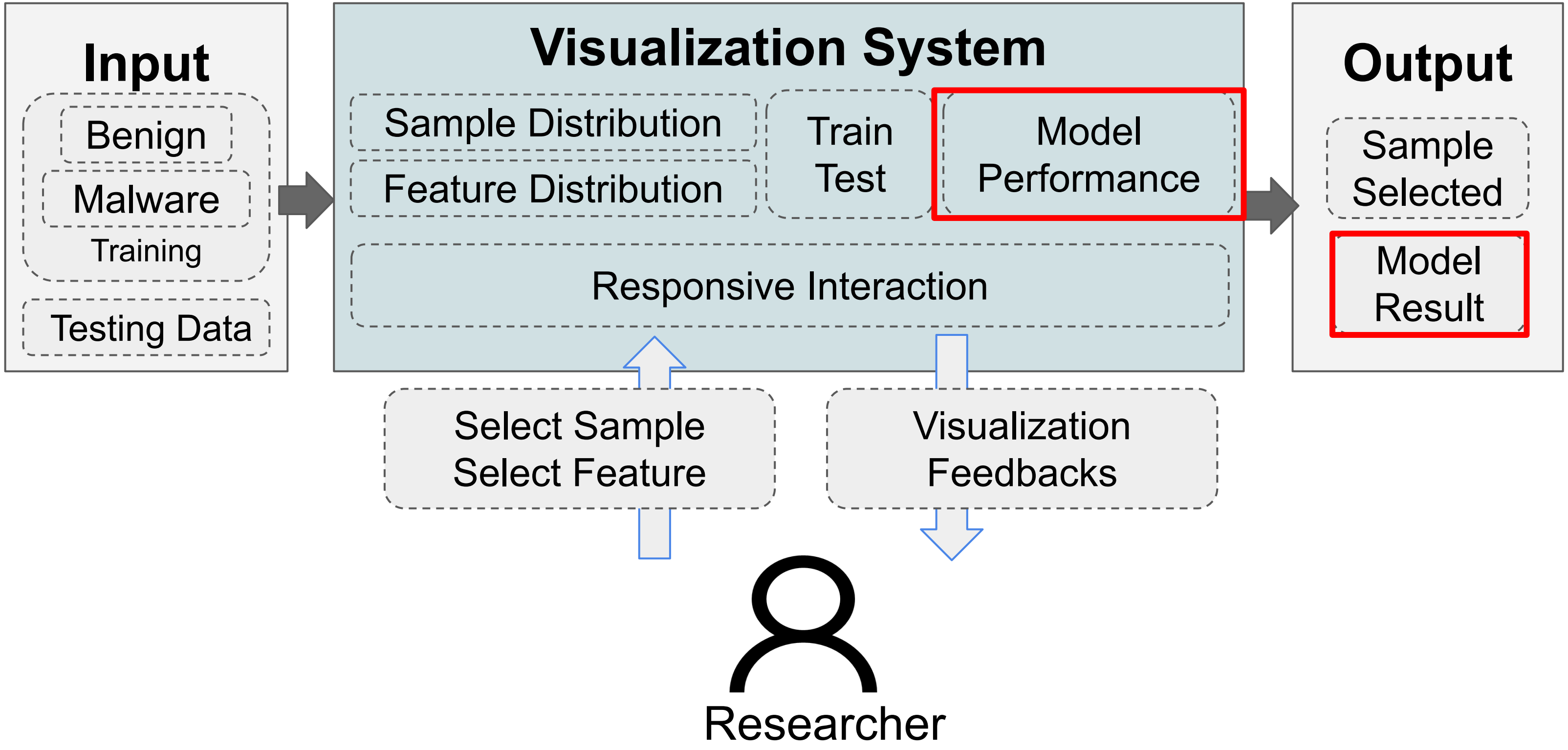
High Level Overview



High Level Overview



High Level Overview

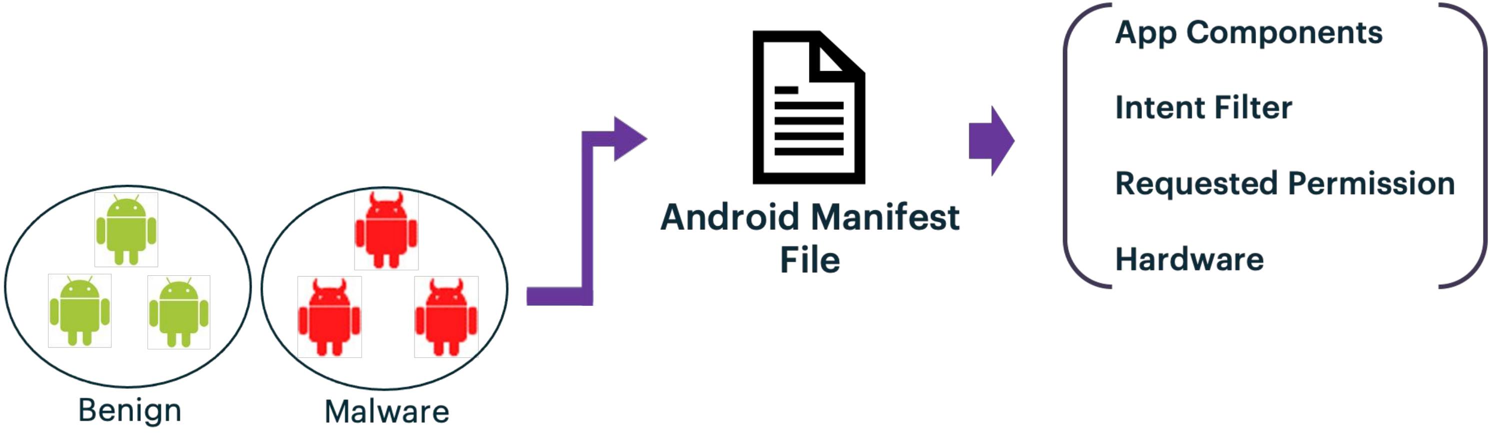


DREBIN: Case Study

- Well-known Android malware detection technique [1]
- Eight categories of features

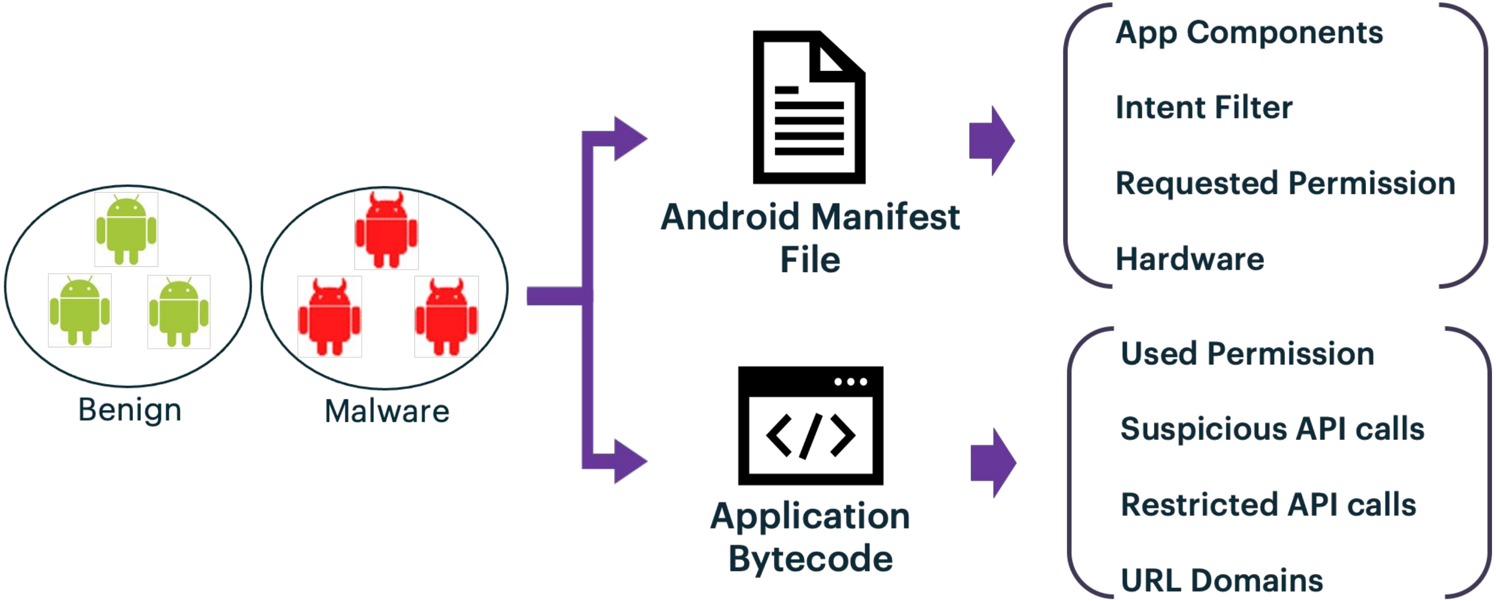
DREBIN: Case Study

- Well-known Android malware detection technique [1]
- Eight categories of features



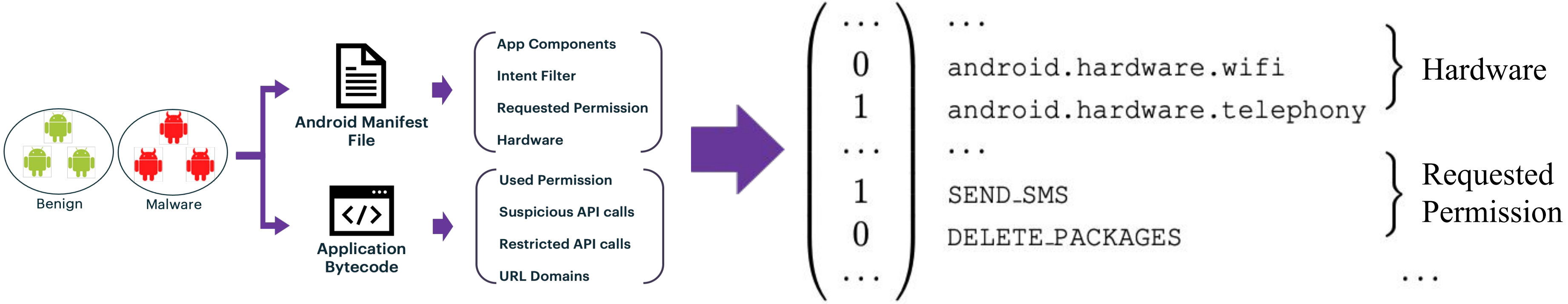
DREBIN: Case Study

- Well-known Android malware detection technique [1]
- Eight categories of features



DREBIN: Case Study

- Binary values indicate presence / absence of feature
- Concatenate features from all samples to form feature space
- SVM classifier



Benign

Malware

Android Manifest File

Application Bytecode

App Components

Intent Filter

Requested Permission

Hardware

Used Permission

Suspicious API calls

Restricted API calls

URL Domains

$$\begin{pmatrix} \cdots \\ 0 \\ 1 \\ \cdots \\ 1 \\ 0 \\ \cdots \end{pmatrix}$$

...

android.hardware.wifi

android.hardware.telephony

...

SEND_SMS

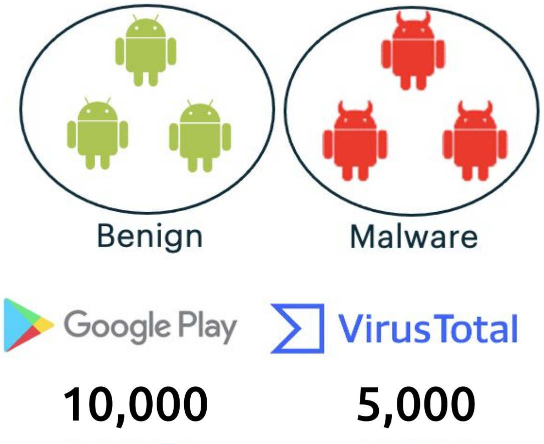
DELETE_PACKAGES

Hardware

Requested Permission

...

Data



Android Sample Data:

- 5,000 malware from VirusTotal
- 10,000 benign from Google Play
- From year 2011 to 2019

Types of Data:

- Android Samples
  - Temporal
  - Drebin Features
- Model
  - Performance
  - Attackability

Tasks (Training)

- Exploring training data distribution

Tasks (Training)

- Exploring training data distribution
    - Compare benign and malware distributions
        - View similarity between samples

Tasks (Training)

- Exploring training data distribution
  - Compare benign and malware distributions
    - View similarity between samples
  - What contribute to the similarity / dissimilarity
    - Individual feature
    - Feature category

Tasks (Training)

- Exploring training data distribution
    - Compare benign and malware distributions
        - View similarity between samples
    - What contribute to the similarity / dissimilarity
        - Individual feature
        - Feature category
    - Observe similarity/dissimilarity by including/discluding features

Tasks (Training)

- Exploring training data distribution
  - Compare benign and malware distributions
    - View similarity between samples
  - What contribute to the similarity / dissimilarity
    - Individual feature
    - Feature category
  - Observe similarity/dissimilarity by including/discluding features

- Select particular set of samples and features to train model

Tasks (Testing)

- Exploring the model
  - Investigate model performance and attackability
    - Accuracy, # features to flip detection

Tasks (Testing)

- Exploring the model
    - Investigate model performance and attackability
        - Accuracy, # features to flip detection
    - Understand important features learned by model
        - Weights of features assigned by the model

Tasks (Testing)

- Exploring the model
    - Investigate model performance and attackability
        - Accuracy, # features to flip detection
    - Understand important features learned by model
        - Weights of features assigned by the model
    - Compare training and testing sample distributions
        - Locate misclassified samples

Tasks (Testing)

- Exploring the model
  - Investigate model performance and attackability
    - Accuracy, # features to flip detection
  - Understand important features learned by model
    - Weights of features assigned by the model
  - Compare training and testing sample distributions
    - Locate misclassified samples
  - Interpret why certain samples being misclassified

Demo

Working scenario of the tool

Limitations

- Limited to analyze “Drebin” Android Malware Detection Tool

Limitations

- Limited to analyze “Drebin” Android Malware Detection Tool

- Limited functionality
    - No way to tune dimensional reduction results

Limitations

- Limited to analyze “Drebin” Android Malware Detection Tool

- Limited functionality
  - No way to tune dimensional reduction results

- Dealing with large feature space
  - Scalability
  - More flexible approaches to select features
  - User has no idea on features for particular samples

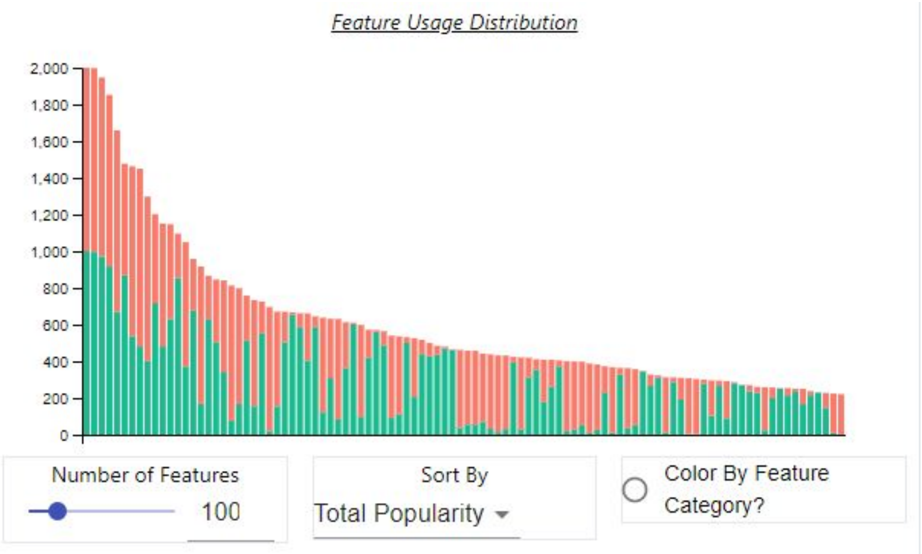
Limitations

- No mechanism for cross experiment comparisons
  - Add juxtaposed view for comparisons
  - Less control over the testing samples

- Few “What-If” functionalities included in the system
  - Allow user to modify / oversample training samples

Lesson Learned

- Prototype, Prototype, Prototype!



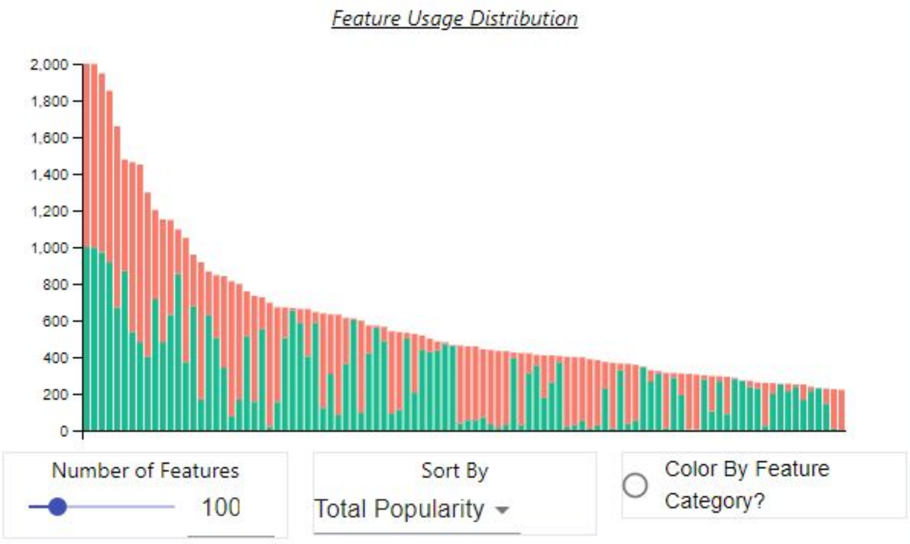
Number of Features 100

Sort By Total Popularity ▾

○ Color By Feature Category?

Lesson Learned

- Prototype, Prototype, Prototype!

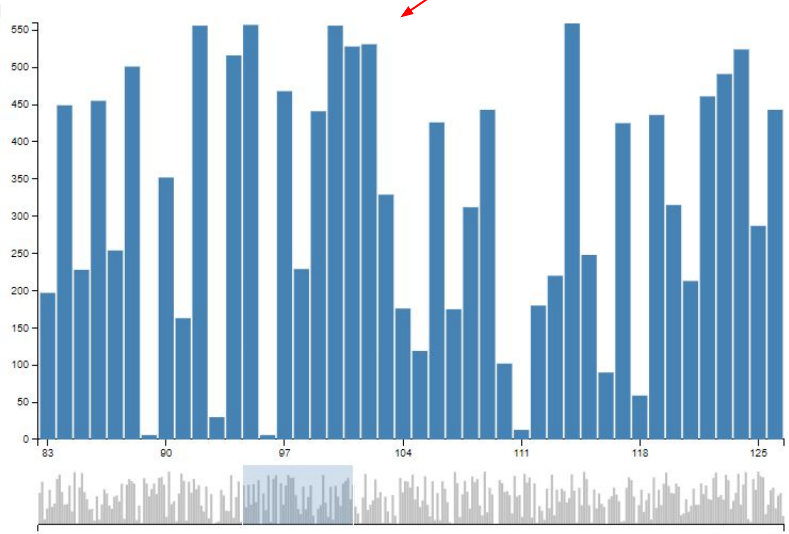


Number of Features 100

Sort By Total Popularity ▾

○ Color By Feature Category?

Focus



Context

Thank you! Q&A

Intro & Framing

- Research Topic (a tool to facilitate exploring the relationship between training data and resulting model)?
  - How to select the set of benign samples that results in the best performance?
  - OR
  - Current process of performing such exploratory tasks are time consuming?